الجامعة الاسمرية الاسلامية
كلية تقنية المعلومات
علوم الحاسوب

العام الجامعي 2019
الفصل الدراسي /
ربيع 2018 / 2019

مقرر/ Internet Programming 2 رمز المقرر/ WT224

المحاضرة الاولى بعنوان

المتغيرات والدوال الشرطية

أ.صالح البوري

نبذة سريعة عن لغة PHP

كلمة ال PHP هي اختصار Pesronal home page tools أي ادوات انشاء الصفحات الشخصية وهذه الادوات انشأها شخص يسمي راسموس وترك هذه الادوات مجانية ومفتوحة المصدر , مما جعل الكثيرين من المبرمجين يشاركون في تطوير هذه الادوات مما جعل هذه اللغة كبيرة وذات اهمية كبيرة

إن كانت لك سابقة عهود مع أي لغة برمجة لن تجد الأمر غريب لأن المنطق البرمجي واحد وأوامرها تشبه إلى حد كبير لغة C . إن كنت من مستخدمي أحد اللغات التالية وهي #C أو ++C أو C أو java ستجد مرونة كبيرة توفرها لك هذه اللغة في التعامل بخلاف ما إعتدت عليه.

أول ما سيصادفك من هذه المرونة أن هذه اللغة لا تحتاج لتعريف متغيرات. فقط إسناد القيمة للمتغير وسيقوم مترجم اللغة بالتعرف على القيمة التي يحويها المتغير تلقائياً. وبما أن صفحة الويب يمكن أن تتضمن أكواد غير أكواد لغة إذاً يجب تنبية المترجم أين أكواد PHP ليتم التعرف عليها ولهذا عند كتابة أكواد PHP داخل الصفحة يجب تضمينها ضمن وسم الفتح php?> ووسم الإغلاق <?

```
<?php
echo ("hello ");
echo "world !!";
?>
```

المتغيرات

في أي لغة برمجة تعتبر المتغيرات وطرق كتابتها من اهم الاساليب البرمجية فدائما نشير للمستخدم كمتغير وكلمة السر كمتغير ، لانه بواسطة المتغيرات نستطيع التحكم ببرنامجنا بيسر وسهولة .

من المميزات الرائعة في لغة php انها آليا تعرف نوع المتغير بل وتقوم بالتحويل بين المتغيرات بكل سهولة فيمكن ان تجعل متغير نص ثم تقوم بتحويله الى رقمي بمجرد كتابة قيمة رقمية به .

انواع المتغيرات المدعومة بلغة php

اولا : الاعداد الصحيحة (Integers) :

هذا النوع من المتغيرات يأخذ القيم الرقمية الصحيحة الموجبة والسالبة مع العلم ان php تعتبر أي رقم بهذه الخصائص وغير محاط بعلامة تنصيص عدد صحيح .

امثلة :

```
$a = 1234;
$a = -1234;
```

ثانيا : الاعداد ذات العلامة العشرية (Floating point numbers) :

هذا النوع من المتغيرات يأخذ القيم الرقمية التي تحتوي على علامات عشرية .

مع العلم ان php تعتبر أي رقم بهذه الخصائص وغير محاط بعلامة تنصيص عدد ذو علامة عشرية .

امثلة :

```
$a = 1.234;
$a = 1.2e3;
```

ثالثا : المتغيرات النصية (string) :

المتغير النصي يأخذ حروف وارقام وعلامات مختلفة و تعتبر لغة php أي قيمة بين علامتي تنصيص على انها متغير نصي .

امثلة :

```
$str = "This is a string 123456789>>>> ";
```

رابعا : المصفوفات (Arrays) :

وهي سلسلة من القيم في متغير واحد يعبر عنه ككائن يحتوى على العديد من القيم ويمكن التعبير عن قيم المصفوفة بالارقام او الاسماء حسب الحاجة

، ومن مهام المصفوفات تسهيل البرمجة وتوفير الوقت والجهد فتستطيع قراءة ملف بجملة واحدة لتجد ان كل سطر في الملف وضع داخل المصفوفة وسوف نتعرف على ذلك عندما نتحدث عن كيفية القراءة من الملفات وكذلك قواعد البيانات وغيرها .
امثلة :

```
$a[0] = "abc";
$a[1] = "def";
$b["foo"] = 13;
```

القفز بين انواع المتغيرات

تقوم لغة php بتغيير نوع المتغير اذا غيرت القيمة التي يحتويها فاذا كانت القيمة رقمية فانه سوف يجعله عدد صحيح واذا كتب بعد ذلك في نفس المتغير قيمة نصية فسوف يحوله آليا الى متغير نصي واليك الامثلة :

```
$a=1 ;  // هذا عدد صحيح
$a=”sager” ;  //الان نص
$a=1 ;  // عاد رقمي
```

كيفية كتابة المتغيرات :

تعتبر لغة الـ php أي كلمة مسبوقة بعلامة ($) على انها متغير اليك المثال التالي :

```
?>php
$var1 = "my ";
$var2 = "program";
echo "$var1 $var2";
<?
```

فلدينا هنا متغيرين في السطر الاول والثاني كل منهما يحمل قيمة وفي السطر الثالث نطلب طباعة هذين المتغيرين لنحصل على الجملة my program .

وهنا يجب ان نأخذ بعين الاعتبار الى ان المتغيرات في الـ PHP لايمكن ان تبدأ برقم ، وايضا ان هناك اختلاف بين الحروف الصغيرة والكبيرة أي s لاتساوي S .

العمليات الحسابية على المتغيرات الرقمية

الصيغة	العملية
$a + $b	جمع متغيرين
$a - $b	طرح متغيرين
$a * $b	ضرب متغيرين
$a / $b	قسمة متغيرين
$a % $b	نسبة المتغير الاول من الثاني
++$a	اضافة واحد الى قيمة المتغير
--$a	طرح واحد من قيمة المتغير

المقارنة بين المتغيرات

الصيغة	العملية
$a == $b	المتغيران متساويان
$a === $b	المتغيران متساويان ومن نفس النوع
$a != $b	المتغيران غير متساويان
$a !== $b	المتغيران غير متساويان او ليسا من النوع نفسه
$a < $b	المتغير الاول اقل من الثاني
$a > $b	المتغير الاول اكبر من الثاني
$a <= $b	المتغير الاول اقل او يساوي الثاني
$a >= $b	المتغير الاول اكبر او يساوي الثاني

الان لنأخذ بعض الامثلة المهمة على المتغيرات :

مثال ١ :

```
$a = 3;
$a += 5;
```

المتغير كان يحمل القيمة ٣ ثم اضفنا اليه ٥ فالناتج يكون ٨ .

نتـيجة : اذن اذا كـنا نـريد ان نجمـع أي قيمة رقمية على متغير نكتب (=+) ثم القيمة .

مثال ٢ :

```
$b = "my ";
$b .= "program";
```

المتغـير كـان يحمـل النص my ثم طلبنا اضافة النص program ليصبح النص الموجود في المتغير هو my program .

نتـيجة : اذن اذا كـنا نـريد ان نضـيف الـى متغير نصـي نكتـب (=.) ثـم النص المراد اضافته .

مثال ٣ :

```
$a = "sager" ;
$$a = "My Name" ;
```

وضـعنا فـي المتغير الاول النص sager وفي السطر الثاني طلبنا من البرنامج ان ينشـأ لن متغير اسـمه النص الموجود في المتغير a ويضع فيه القيمة My Name أي اصـبح لديـنا متغيرين هما $a و $sager كما لو اننا كتبنا الجملة التالية :

```
$a = "sager" ;
$sager = "My Name" ;
```

نسـتخدم هـذه الطـريقة عـندما نريد ان نربط متغير معين بمتغيرات اخرى مثل الاسـم وكلمة السر .

كيف نضيف تعليق

عـندما نكتـب بـرنامج ونعـود الـيه بعد مدة نجد اننا ننسى الكثير من الخطوات التـي قمـنا بهـا او لمـاذا كتبـنا هـذا الامر او ذاك ، او عمل هذه الدالة او تلك ، بالإضـافة الـى ان هـذا الـبرنامج قـد يقـوم اشخاص آخرين بالتعديل عليه لذلك تعتـبر التعلـيقات مـن الامـور التـي تبيـن ان المـبرمج محـترف وعملـه منظم ، ولكتابة التعليقات نتبع التالي :

لاجل كتابة تعليق من سطر واحد نستخدم // ثم التعليق الذي نريد
مثال :

```
؟>php
هذه ملاحظه //
Echo "line1" ;
ملاحظة اخرى //
Echo "line2" ; // ملاحظة خلف سطر امر مباشره
<؟
```

بيـنما لـو اردنـا ان نكتـب تعلـيق مـن عـدة اسطر فاننا نستخدم */ في بداية التعليق ونسـتخدم /* لنخبر البرنامج ان ملاحظاتنا انتهت .
مثال :

```
؟>php
Echo "my program" ;
/*
السطر الاول من الملاحظات
السطر الثاني
السطر الثالث من الملاحضات
*/
<؟
```

وكمـا تلاحظ اننا نستطيع ان نكتب التعليقات في أي مكان من البرنامج ، فكل ماعلينا فعله هو اخبار البرنامج ان هذا السطر او السطر التالي ملاحظة .

المسارات في php

عندما نتحدث عن ملف معين فإننا نذكر انه يجب وضع المسار الكامل له خصوصا اذا لم يكن في نفس المجلد الموجود به البرنامج الذي كتبته ، وعموما هناك عدة حالات لمسار الملف بحيث تتعرف عليه الدوال المختلفة :

لنفترض اننا نود التعامل مع الملف "file.txt" فيكون لدينا احد الحالات التالية :

الحالة الأولى : ان يكون في نفس المجلد الذي ننفذ فيه برنامجنا وهنا نستطيع كتابة اسم الملف فقط دون الحاجة الى إلية إضافات مثل "file.txt" .

الحالة الثانية : ان يكون الملف في مجلد اسمه data داخل المجلد الموجود به برنامجنا فعند هذه الحالة نكتب مسار الملف كما يلي :

"./data/file.txt"

الحالة الثالثة : ان يكون الملف في المجلد الاب للمجلد الموجود به برنامجنا ، وهنا يصبح مسار الملف كما يلي :

"../file.txt"

الحالة الرابعة : ان نحدد المسار كعنوان انترنت مثل " www.4alarab.com/file.txt "

عموما يجب عليك الا تقلق فهذه المفاهيم العامة سوف تتضح بشكل كبير وسوف تجدها سهلة جدا عندما نتقدم قليلا ونتحدث عن الملفات وكيفية التعامل معها سواء فتح الملفات للقراءة او الكتابة او انشاء الملفات ، لذلك اتوقع منك فقط ان تتعرف فقط على ان مسار الملف مهم كأهمية ان تصف الطريق جيدا لم تريد ان يصل الى موقع محدد ، دون ان يضطر الى سلك الطرق الخاطئة وقد لا يصل ابدا .

اتخاذ القرارات

في اثناء كتابتك للبرنامج سوف تلاحظ انك بحاجة الى اتخاذ قرار معين ، مثلا هل اسمح للزائر بالدخول الى صفحة معينة او الاطلاع على معلومات محددة ؟ وبالطبع فان الاجابة على هذا السؤال تعتمد على تحقق شروط معينة مثل هل يملك هذا الشخص صلاحيات لدخول الموقع وهل كلمة السر التي ادخلها صحيحة ؟

وهكذا نحن بحاجة الى اتخاذ مثل هذه القرارات دائما ولكي نتخذ القرار يجب ان نتأكد من ان الشرط يتحقق مثلا هل كلمة السر صحيحة ، فاذا كانت كلمة السر صحيحة يعيد لنا البرنامج القيمة true بينما اذا لم تكن صحيحة فسيعيد لنا القيمة false .

اذن الشرط هو عبارة عن مقارنة بين القيمة التي يدخلها المستخدم والقيمة الموجودة لدينا فاذا كانت القيمتان متساويتان فيعني انه ادخل كلمة السر الصحيحة وهذا يعني انه يمكنه الدخول الى الموقع .

الدالة الشرطية IF (اذا كان فـ)

في هذه الدالة الشرطية نقوم بالتأكد من الشرط وسوف يكون لدينا هنا حالتان ام ان يتحقق الشرط وهنا سوف يتم الدخول الى الدالة وتنفيذ العمليات الموجودة بداخلها ثم الانتقال وتنفيذ الاوامر التي تقع خارجها ، او ان لا يتحقق الشرط وهنا سوف يتم الانتقال مباشرة الى نهاية الدالة الشرطية وتنفيذ الاوامر التي تقع خارجها .

صيغتها : وهو عبارة عن مقارنة بين القيمة

```
if ( الشرط ) {
نفذ هذا اذا تحقق الشرط
}
أي اوامر خارج الدالة الشرطية
```

مثال :

```
?>php
$a="pass" ;
if ($a=="pass" ) {
echo $a ;
}
print "End" ;
<?
```

في هذا المثال نختبر القيمة الموجوده في المتغير $a فاذا كانت تساوي pass نقوم بطباعتها ثم ننتقل الى خارج دالة القرار ، اما اذا كانت لا تساوي هذه القيمة فاننا ننتقل مباشرة الى خارج دالة القرار.

تلاحظ هنا اننا ننفذ اوامر معينه فقط عندما يتحقق الشرط .

الدالة الشرطية IF else (اذا كان فـ واذا لم يكن فـ)

فـي هـذه الدالـة الشـرطية نقـوم بالـتأكد مـن الشـرط وسـوف يكون لدينا هنا حالـتان ام ان يـتحقق الشـرط وهـنا سـوف يـتم الدخـول الـى الدالـة وتنفـيذ العملـيات الموجـودة بداخلهـا ثـم الانتقال وتنفيذ الاوامر التي تقع خارجها ، او ان لا يـتحقق الشـرط وهنا سوف يتم تنفيذ الاوامر الموجودة بعد الكلمة else ثم الانتقال مباشـرة الى نهاية الدالة الشرطية وتنفيذ الاوامر التي تقع خارجها .

صيغتها :

```
if ( } ( الشـرط
نفذ هذا اذا تحقق الشـرط
} else {
نفذ هذا اذا لم يتحقق الشـرط
}
```

مثال :

```
?>php
if ($a=="sager" ) {
echo $a ;
} else {
echo " ; "كلمة السر غير صحيحة
}
print “end” ;
<?
```

فـي هـذا المـثال نختـبر القـيمة الموجـوده في المتغير $a فاذا كانت تساوي sager نقـوم بطباعـتها ثم ننتقل الى خارج دالة القرار ، اما اذا كانت لا تساوي هـذه القـيمة فانـنا نـنتقل الـى الخـيار الـبديل الذي يأتي بعد الكلمة else ثم ننتقل الى خارج دالة القرار .

تلاحـظ هنا اننا ننفذ اوامر معينه عندما يتحقق الشـرط ، وننفذ ايضا اوامر معينه اذا لم يتحقق الشـرط .

الدالة الشرطية elseif (اذا كان فـ او اذا كان فـ اواذا لم يكن فـ)

فـي هـذه الدالـة الشـرطية نقـوم بالـتأكد مـن الشـرط وسـوف يكون لدينا هنا حالـتان ام ان يـتحقق الشـرط وهـنا سـوف يـتم الدخـول الـى الدالـة وتنفـيذ العملـيات الموجـودة بداخلها ثـم الانتقال وتنفيذ الاوامر التي تقع خارجها ، او ان لا يـتحقق الشـرط وهـنا سـوف يـتم التأكد من الشـرط الثاني وهكذا الثالث واذا لم لم تتحقق الشروط جميعا يتم تنفيذ الاوامر الموجودة بعد الكلمة else ثم الانتقال مباشرة الى نهاية الدالة الشرطية وتنفيذ الاوامر التي تقع خارجها .

صيغتها :

```
if () { الشرط الاول )
نفذ هذا اذا تحقق الشرط الاول
} elseif () { الشرط الثاني )
نفذ هذا اذا تحقق الشرط الثاني
} else {
نفذ هذا اذا لم يتحقق أي من الشروط
}
```

مثال :

```
?>php
if ($a=="sager" ) {
echo "اهلا صقر" ؛
} elseif ($a=="khaled") {
echo "اهلا خالد" ؛
} else {
echo " الاسم غير معروف" ؛
}
<?
```

فـي هـذا المـثال نختـبر القـيمة الموجـوده في المتغير $a فاذا كانت تساوي sager نقوم بطباعة رسالة الترحيب اهلا صقر ثم ننتقل الى خارج دالة القرار ، بيـنما اذا كانـت القـيمة تسـاوي khaled فاننا نطبع رسالة الترحيب اهلا خالد ثـم نـنـتقل الـى خـارج دالـة القـرار ، امـا اذا كانـت القـيمة لاتسـاوي جمـيع الاحـتمالات فانـنا ننتقل الى الخيار البديل الذي يأتي بعد الكلمة else ونطبع الجملة الاسـم غير معروف ثم ننتقل الى خارج دالة القرار .

تلاحـظ هـنا انـنا نـنفذ اوامـر معيـنه عندما يتحقق احد الشروط او الاحتمالات ، ونـنفذ ايضا اوامـر معيـنه اذا لـم يـتحقق أي مـن هذه الشروط او الاحتمالات .

حلقة التكرار while

استمر في حلقة مادام الشرط متحققا ويتم التحقق من الشرط قبل الدخول الى الحلقة في كل مرة ، فعند بداية الحلقة نتأكد من الشرط فاذا تحقق نقوم بالدخول الى الحلقة وتنفيذ الاوامر مرة واحدة ثم نعود لرأس الحلقة مرة اخرى ونتأكد من الشرط وهكذا يستمر البرنامج حتى يصبح الشرط غير متحقق فيقوم بالخروج من الحلقة وتنفيذ الاوامر الموجودة خارجها .

وتساعد الحلقات كثيرا في اختصار الاوامر التي تكتب فلو اردنا ان نحصل على مائة سجل من قاعدة البيانات وطباعتها فاذا لم نستخدم الحلقة فسنحتاج لمئات الاوامر ولكن بواسطة الحلقة المناسبة فلن نكتب اكثر من عشرة اسطر .

صيغتها :

```
while ( الشرط ):
نفذ دائما مادام الشرط متحققا
endwhile;
```

مثال :

```
?>php
$a=0 ;
while ($a <18 ):
++$a ;
echo $a ;
endwhile;
<?
```

في هذا المثال سوف نرى مفعول الحلقة في اختصار كتابة الاوامر ففي هذا المثال البسيط نحن بحاجة الى طباعة الارقام من ١ الى ١٧ فلو استخدمنا الطريقة العادية فسوف نحتاج الى ١٧ سطر يحتوي على جملة echo يليها المتغير الذي نود طباعته وبالاضافة الى زيادة الكوود يكون من الصعب صيانة البرنامج او تطويره .

فلو اردنا مثلا طباعة الارقام حتى ١٢٠ فكل ماعلينا هو تغيير الرقم الموجود في السطر الثالث من ١٨ الى ١٢٠ وهكذا .

لا حظ ان البرنامج يتحقق من صحة الشرط قبل دخوله للحلقة ومن ثم فاننا لن نطبع القيمة ١٨ لانها تخالف الشرط .

الجامعة الاسمرية الاسلامية
كلية تقنية المعلومات
علوم الحاسوب

الفصل الدراسي /
خريف 2019 / 2020

مقرر/ Internet Programming 2 رمز المقرر/ CS220

محاضرة بعنوان :

إرسال البيانات عبر النماذج والجلسات

أ.صالح البوري

الدرس العاشر: تمرير المتغيرات من خلال عنوان الموقع (URL)

عندما تبرمج بلغة PHP يمكنك أن تمرر المتغيرات من صفحة لأخرى، في هذا الدرس حول تمرير المتغيرات من خلال عنوان الموقع.

كيف تعمل؟

لعلك تساءلت لم بعض عناوين المواقع تبدو بهذا الشكل:

```
http://html.net/page.php?id=1254
```

لم هناك علامة استفهام بعد اسم الصفحة؟

الإجابة هي أن الأحرف بعد علامة الاستفهام هي سلسلة استعلام**HTTP** (سلسلة استعلام HTTP يمكنها أن تحوي متغيرات وقيم) في المثال أعلاه السلسلة تحوي متغيراً اسمه "id" وقيمته "١٢٥٤".

هذا مثال آخر:

```
http://html.net/page.php?name=Joe
```

مرة أخرى، هناك متغير اسم "name" وقيمته."Joe"

كيف تحصل على المتغير من خلالPHP ؟

لنفترض أن لديك صفحة PHP اسمها**people.php** ، الآن يمكنك أن تطلب هذه الصفحة من خلال العنوان التالي:

```
people.php?name=Joe
```

باستخدامPHP ، يمكنك الحصول على قيمة المتغير 'name' بهذا الأمر:

```
$_GET["name"]
```

أنت تستخدم أمر $_GET لتجد قيمة متغير محدد، لنجربه في مثال:

```
<html>
<head>
<title>Query string</title>
</head>
<body>

<?php

// The value of the variable name is found
echo "<h1>Hello " . $_GET["name"] . "</h1>";

?>

</body>
</html>
```

شاهد المثال(keep an eye on the URL)

عندما تطلع على المثال أعلاه جرب أن تغير الاسم "Joe" بإسمك في عنوان الموقع واطلب الصفحة مرة أخرى.

متغيرات عديدة في نفس العنوان

يمكنك تمرير أكثر من متغير في عنوان الموقع، بفصل المتغيرات باستخدام حرف **&** يمكن تمرير متغيرات متعددة:

```
people.php?name=Joe&age=24
```

في عناوين الموقع هناك متغيرين، nameوage، بنفس الطريقة أعلاه يمكنك الحصول على قيمتهما بهذا الشكل:

```
$_GET["name"]
$_GET["age"]
```

لنضف متغيراً آخراً للمثال:

```
<html>
<head>
<title>Query string </title>
</head>
<body>

<?php

// The value of the variable name is found
echo "<h1>Hello " . $_GET["name"] . "</h1>";

// The value of the variable age is found
echo "<h1>You are " . $_GET["age"] . " years old </h1>";

?>

</body>
</html>
```

شاهد المثال(keep an eye on the URL)

الآن تعلمت طريقة واحدة لتمرير القيم بين الصفحات باستخدام عنوان الموقع، في الدرس التالي سنلقي نظرة على طريقة ثانية، النماذج.

الدرس الحادي عشر: تغيير المتغيرات من خلال النماذج

المواقع التفاعلية تتطلب مدخلات من المستخدمين، أحد أكثر الطرق استخداماً للحصول على المدخلات هي النماذج.

في هذا الدرس سنلقي نظرة على كيفية إنشاء نموذج ومعالجة مدخلاته في المزود.

<form>

عندما تبرمج نموذجاً فهناك خاصيتان مهمتان: **action** و. **method**

action

تستخدم لإدخال عنوان الصحفة التي ستسقبل مدخلات النموذج، ستكون ملف PHP تريده أن يتعامل مع المدخلات.

method

ما الفرق بين Post و Get ؟

يمكنها أن تقبل إحدى القيمتين إما "post" أو"get" ، وهما طريقتان مختلفتان لتمرير المدخلات، في هذه المرحلة لست بحاجة لمعرفة الكثير، باستخدام "get" البيانات ترسل من خلال عنوان الموقع، وباستخدام "post" البيانات ترسل من خلال كتلة بيانات من خلال تقنية(STDIN)، في بقية الدرس سنلقي نظرة على كيفية الحصول على البيانات من خلال عنوان الموقع باستخدام$_GET ، في هذا الدرس سنلقي نظرة على كيفية استرجاع البيانات المرسلة بطريقة."post"

صفحة HTML تحوي نموذجاً

الصفحة التي تحوي نموذجاً لا تحتاج لأن تكون ملفPHP ، يمكنها أن تكون إن أردت، ولا تحتاج حتى لأن تكون في نفس الموقع الذي يحوي ملفاً سيتلقى البيانات.

في المثال الأول سنلقي نظرة على نموج بسيط يحوي حقلاً نصياً واحداً::

```
<html>
<head>
<title>Form</title>
</head>
<body>

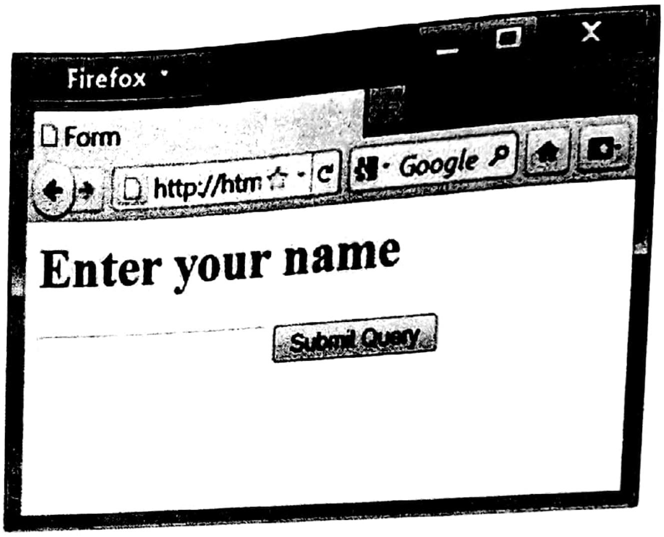
<h1>Enter your name</h1>

<form method="post" action="handler.php">
<input type="text" name="username">
<input type="submit">
```

```
</form>

</body>
</html>
```

النتيجة في المتصفح ستكون:



هنا يأتي الجزء الممتع: تلقي البيانات ومعالجتها باستخدام.PHP

طلب بيانات النموذج باستخدامPHP

عندما تحتاج لطلب بيانات أرسلت من خلال نموذج بطريقة post فعليك استخدام أمر $_POST:

```
$_POST["fieldname"];
```

الذي يأخذ قيمة حقل في النموذج، لنجرب استخدامه في مثال.

في البداية اكتب صفحة تحوي النموذج كما في المثال أعلاه، ثم اكتب صفحة PHP وسمها "handler.php"لاحظ أن هذا هو اسم الصفحة الذي كتبناه في خاصية action في النموذج.

ملف "handler.php" سيحوي التالي:

```
<html>
```

```
<head>
<title>Form</title>
</head>

<body>

<?php

echo "<h1>Hello " . $_POST["username"] .
"</h1>";

?>

</body>
</html>
```

شاهد المثال

مدخلات المستخدم والشروط

في المثال التالي سنجرب استخدام مدخلات المستخدم لإنشاء شروط، في البداية نحن بحاجة لنموذج:

```
<html>
<head>
<title>Form</title>
</head>
<body>

<form method="post" action="handler.php">

<p>What is your name:</p>
<input type="text" name="username"></p>

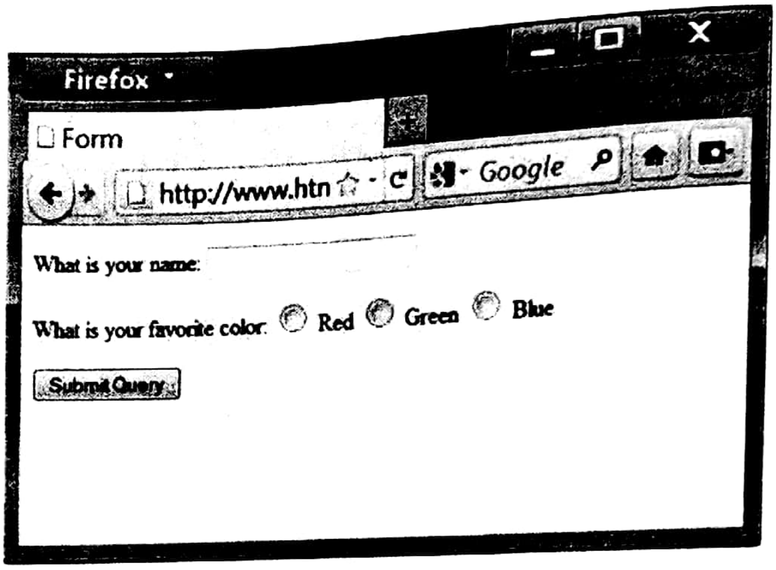
<p>What is your favorite color:</p>
<input type="radio" name="favoritecolor"
value="r" />Red
<input type="radio" name="favoritecolor"
value="g" />Green
<input type="radio" name="favoritecolor"
value="b" />Blue </p>

<input type="submit" value="Submit" />
```

```
</form>

</body>
</html>
```

سيبدو بهذا الشكل في المتصفح:



الآن سنستخدم هذه المدخلات لإنشاء صفحة تغير لون خلفيتها تلقائياً بحسب اختيار المستخدم للونه المفضل، يمكننا فعل ذلك من خلال إنشاء شرط، (اقرأ الدرس السادس (يستخدم البيانات التي أدخلها المستخدم في النموذج.

```
<?php

$strHeading = "<h1>Hello " .
$_POST["username"] . "</h1>";

switch ($_POST["favoritecolor"]) {
case "r":
      $strBackgroundColor = "rgb(255,0,0)";
      break;
case "g";
      $strBackgroundColor = "rgb(0,255,0)";
      break;
case "b":
      $strBackgroundColor = "rgb(0,0,255)";
      break;
default:
```

```
            $strBackgroundColor = "rgb(255,255,255)";
            break;
    }

    ?>

    <html>
    <head>
    <title>Form</title>

    </head>
    <body      style="background:           <?php       echo
$strBackgroundColor; ?>;">

    <? echo $strHeading; ?>

    </body>
    </html>
```

لون خلفية الصفحة سيكون أبيضاً إن لم يختر المستخدم لوناً مفضلاً في النموذج، هذا يمكن إنجازه من خلال أمر **default** لتحديد ما يحدث إن لم تتحقق أي من الشروط السابقة.

لكن ماذا لو لم يدخل المستخدم اسمه؟ البرنامج سيقول فقط ,"Hello" سنضيف شرطاً إضافياً لتغيير هذا.

```
    <?php

    $strUsername = $_POST["username"];

    if ($strUsername != "") {
        $strHeading    =      "<h1>Hello      "      .
$_POST["username"] . "</h1>";
    }
    else {
        $strHeading = "<h1>Hello stranger!</h1> ";
    }

    switch ($_POST["favorite color"]) {
    case "r":
            $strBackgroundColor = "rgb(255,0,0)";
            break;
```

```
case "g";
      $strBackgroundColor = "rgb(0,255,0)";
      break;
case "b":
      $strBackgroundColor = "rgb(0,0,255)";
      break;
default:
      $strBackgroundColor = "rgb(255,255,255)";
      break;
}

?>

<html>

<head>

<title>Form</title>
</head>
<body style="background: <?php echo
$strBackgroundColor; ?>;">

<? echo $strHeading; ?>

</body>
</html>
```

شاهد المثال.

في المثال أعلاه استخدمنا شرطاً **للتحقق** من المعلومات التي أرسلها المستخدم، في هذه الحالة قد لا يكون من المهم أن يكتب المستخدم اسمه، لكن عندما تبدأ في برمجة برامج متقدمة من الضروري أن تضع في الحسبان أن المستخدم قد لا يدخل المعلومات كما ترغب.

مثال: نموذج مراسلة

بمعرفتك الجديدة حول لغة PHP والنماذج يمكنك إنشاء نموذج مراسلة من خلال استخدام الوظيفة mail ، التي تكتب بهذه الطريقة:

```
mail(to, subject, message);
```

أولاً نحن بحاجة لنموذج بسيط:

```
<html>
<head>
<title>Contact form</title>
</head>
<body>

<h1>Contact form</h1>

<form method="post" action="handler.php">
<p>Subject:<br /><input type="text"
name="subject" /></p>
<p>Message:<br /><textarea
name="message"></textarea></p>
<input type="submit">
</form>

</body>
</html>
```

ثم نحن بحاجة إلى برنامج PHP ليرسل مدخلات المستخدم:

```
<html>
<head>
<title>Functions</title>
</head>
<body>

<?php

// Recipient (change to your e-mail address)
$strEmail = "name@mydomain.com";

// Get user inputs
$strSubject = $_POST["subject"];
$strMessage = $_POST["message"];

mail($strEmail,$strSubject,$strMessage);
echo "Mail Sent.";
```

```
?>

</body>
</html>
```

لاحظ أن المثال سيعمل فقط إن كنت تملك حق الوصول إلى مزود بريد إلكتروني، هذا غير متوفر في XAMPP ومعظم خدمات الاستضافة المجانية، كذلك وبعض شركات الاستضافة تطلب إضافة معلومات المرسل والذي يمكن إضافته بهذا الشكل:

```
mail("you@yourdomain.com", "Test", "This is a test mail", "From: me@mydomain.com");
```

ماهي الجلسات PHP Sessions

الجلسات PHP Sessions هي عبارة عن طريق تجعل البيانات قابلة للوصول اليها من خلال عدة صفحات على مستوى الموقع بالكامل.بمعنى تمنحك الصلاحية للدخول الى المعلومات على صفحات الويب.
الجلسات PHP Sessions تقوم بانشاء ملف في المجلد - المسار المؤقت على السيرفر حيث يتم تسجيل متغيرات الجلسة فيها وقيمها.وسوف تكون البيانات متوفر لكل الصفحات التي على موقع عند زيارة الصفحات.
مكان ملف الجلسات PHP Sessions المؤقت يتم تحديده من خلال ملف الاعدادات php.ini في السطر **session.save_path**
وقبل تحديد المسار يجب التاكد من وجود الملف وصحة مساره.

عند بدء الجلسات PHP Sessions تحدث الخطوات التالية

- تقوم لغة البي اتش بي بانشاء معرف نصي مميز طوله 32 من سداسي عشر مثل c7foj34c3jj973hjkop2fc937e34433
- تقوم الكوكيز باستدعاء **PHPSESSID** التى ارسلت الى جهاز المستخدم ليحفظ معرف الجلسة المميز كما في الخطوة السابقة.
- يتم انشاء ملف في مجلد الجلسات المؤقت يحمل نفس القيمة التي ارسلت للجهاز المستخدم.

عندما يريد سكريبت PHP استرداد القيمة من متغير الجلسة ال معرPHP تلقائيا تحصل على قيمة معرف الجلسة المميز من الكوكيز PHPSESSID ثم تبحث في الدليل الخاص ب مجلد الجلسات المؤقت عن الملف الذي يحمل نفس القيمة والتحقق من صحة القيم من خلال مقارنة القيمتين.
الجلسة تنتهي عندما يفقد المستخدم المتصفح أو بعد مغادرة الموقع، سيقوم السيرفر بإنهاء الجلسة بعد مرور فترة محددة من الزمن، عادة 30 دقيقة .

بدء الجلسة في البي اتش بي Starting a PHP Session:

الجلسة تبدء عن طريق استدعاء الدالة() session_start تقوم هذه الدالة بفحص هل الجلسة بدات ام لا في حال لم تبداء تقوم ببدء جلسة جديدة. يفضل استخدام هذه الدالة في بداية ملف البي اتش بي.
يتم حفظ المتغيرات وقيمها في المصفوفة المساعدة[] **$_SESSION** ويمكن الوصول الهذه المتغيرات خلال فترة حياة الجلسة.

مثال على الجلسات في البي اتش بي

في المثال التالي سيتم حفظ المتغيرات الجلسة في counter سنتخدم الدالة **isset()** لفحص هل تم حفظ المتغيرات في المصفوفة المساعدة ام لا

```
<?php
   session_start();
   if( isset( $_SESSION['counter'] ) )
   {
      $_SESSION['counter'] += 1;
   }
   else
   {
      $_SESSION['counter'] = 1;
   }
   $msg = "You have visited this page ".
$_SESSION['counter'];
   ?>
<html>
<head>
<title>بدء الجلسة</title>
</head>
<body>
<?php  echo ( $msg ); ?>
</body>
</html>
```

إنهاء أو تدمير الجلسة DESTROYING A PHP SESSION:

كما توجد دالة لبدء الجلسة ايضا توجد دالة لانهاء او ازالة او تدمير الجلسة **session_destroy()**

الجامعة الاسمرية الاسلامية
كلية تقنية المعلومات
علوم الحاسوب

الفصل الدراسي /
خريف 2019 / 2020

مقرر/ Internet Programming 2

محاضرة بعنوان :

المصفوفات (Array) و الوظائف (Function)

أ.صالح البوري

المصفوفات Array في لغةPHP

المصفوفات عبارة عن متغيرات تحمل أكثر من قيمة وأكثر من نوع بيانات، ويتم أستخدام المصفوفات لتوفير الوقت والمجهود الغير ضروري، وذلك لأن المصفوفات توفر عليك أنشاء أكثر من متغير عن طريق تخزين أكثر من قيمة واحدة في المصفوفة، ولا تقتصر وظيفة المصفوفات علي أختصار الوقت فقط بل أختصار عدد الشفرات البرمجية وهذا يساعد علي سهولة العمل وتفادي التشتت الذي يحدث نتيجة العدد الكبير من المتغيرات.

- نستخدم المصفوفات لأختصار الوقت
- المصفوفات تحمل أكثر من قيمة وأكثر من نوع بيانات
- المصفوفات تختصر الأسطر البرمجية

ويمكنك أن تتخيل في حالة تسجيل كل أسماء دول العالم في متغيرات، في تلك الحالة سوف يكون هناك عدد هائل جداً من المتغيرات التي كل متغير فيها يحمل أسم دولة، لأن المتغيرات لا تحمل إلا قيمة واحدة ، وسيكون لدينا متغيرات بعدد دول العالم .

والأن تخيل معي بأنك قادر علي أختصار هذا العدد الهائل من المتغيرات إلي متغير واحد يحمل كل العناصر، عن طريق المصفوفات .

```
<?php
$color = array ("red","green","blue","yellow");
?>
```

أنواع المصفوفات

المصفوفات تنقسم إلي العديد من الأنواع، ولا يختلف كل نوع عن الأخر إلا أختلاف بسيط جداً، بحيث تجد مصفوفات يتم ترتيبها عن طريق قيم عددية، ومصفوفات أخري يتم ترتيبها عن طريق قيم نصية أو (مفاتيح) ، والنوع الأخير هي مصفوفات متداخلة مع بعضها .

- indexed Array : هي مصفوفة يتم ترتيبها عن طريق قيم عددية
- Associative Array : هي مصفوفة يتم ترتيبها عن طريق (مفاتيح)
- Multidimensional Arrays : مصفوفات متداخلة

المصفوفات الرقمية Indexed Array

المصفوفات الرقمية indexed Array هي النوع الأول من أنواع المصفوفات، وما يميز هذة المصفوفة بأنه يتم التعبير عن كل خانة في المصفوفة برقم ثابت يبدأ من الصفر لأول خانة ومن ثم يتصاعد الرقم مع عدد الخانات في المصفوفة .

يمكننا أن نفترض بأنك قمت بتخزين 3 عناصر في مصفوفة، الأن كيف يمكننا أن نتحكم في كل عنصرر علي حدى ؟ ,مثل طباعتها او التعديل عليها او حتي أزالتها من المصفوفة، كل ذلك يكون عن طريق الترتيب الأساسي لكل قيمة في المصفوفة .

- القيمة الأولي = 0
- القيمة الثانية = 1
- القيمة الثالثة = 2

```
<?php
$color = array("red","green","blue");
echo $color[0]; // red
echo $color[1]; // green
echo $color[2]; // blue
?>
```

في المثال السابق قمنا بأنشاء مصفوفة تحمل ثلاثة عناصر ("احمر"،"أخضر"،"أزرق")، وكل عنصر لة رقم ثابت للتحكم فيه، ومع أستخدام دالة echo , قمنا بطباعة كل عنصر علي حدي بأستخدام أسم $المتغير مع [رقم كل عنصر في المصفوفة] .

- اللون الأحمر = [0]
- اللون الأخضر = [1]
- اللون الأزرق = [2]

تعديل عنصر : يمكنك تعديل عنصر في المصفوفة وذلك عن طريق معرفة رقم العنصر الذي تريد التعديل علية، ومن ثم أستخدام علامة (=) لتعيين قيمة جديدة بدلاً عن القيمة القديمة .

$المصفوفة [رقم العنصر] = القيمة الجديدة

```
<?php
$color = array("red","green","blue");
$color[0] = "black"; // change red to black
?>
```

اضافة قيمة جديدة: يمكنك أضافة قيمة جديدة إلي المصفوفة، ويوجد نوعين من الأضافات في مصفوفات لغة php , الطريقة الأولي هي طريق تلقائية تقوم بأضافة العنصر في نهاية المصفوفة، والطريقة الثانية عن طريق تحديد مكان محدد للعنصر في المصفوفة بواسطة الترتيب الرقمي .

- اولاً : أضافة تلقائية في نهاية المصفوفة,
- ثانياً : أضافة في مكان محدد من أختيارك .

الأضافة التلقائية :

اضافة عنصر بشكل تلقائي إلي المصفوفة عن طريق تحديد المصفوفة مع الأقواس المربعة [ ]، ومن ثم أستخدام علامة (=) لتعيين قيمة العنصر الجدّيد

```
<?php
$color = array("red","green","blue");
$color[] = "maginta"; // أضافة قيمة جديدة إلي المصفوفة
?>
```

الأضافة المرتبة :

لأضافة عنصر في مكان محدد في المصفوفة يجب تحديد مكان العنصر اولاً ومن ثم تحديد ترتيب العنصر وكتابة الترتيب بين الأقواس المربعة [الترتيب]

```
<?php
$color = array("red","green","blue");
$color[5] = "maginta"; // أضافة قيمة جديد تحمل الرقم 5
?>
```

مصفوفة المفاتيح Associative Array

تستخدم مصفوفة المفاتيح، لتحديد قيمة نصية يتم معاملتها كمفتاح للقيمة الأساسية، ولنفترض باننا نقوم بأنشاء مصفوفة تحمل ثلاثة عناصر مثل (أسم شخص، وعنوانة، وعمرة)، في تلك الحالة يمكننا أستخدام مفتاح باسم Name من أجل اسم

الشخص، ومفتاح باسم Adress لعنوان الشخص، ومفتاح باسم Age لعمر الشخص، وعند استدعاء القيمة يتم أستدعائها عن طريق المفاتيح .

1. يتم الفصل بين المفتاح وقيمتة بعلامة(=>)
2. أسم المفتاح يكون من أختيارك

```
<?php
$person = array("name" => "Mohamed","adress" => "green","age" => "blue");
echo $person["name"];
echo $person["adress"];
echo $person["age"];?>
```

تعديل عنصر : يمكنك تعديل العنصر عن طريق أستخدام مفتاح كل عنصر تريد التعديل عليه، ومن ثم أستخدام علامة (=) لتعيين القيمة الجديدة بدلا من القيمة القديمة .

```
<?php
$person = array("name" => "Mohamed","adress" => "green","age" => "blue");
$person["name"] = "Ahmed" ; // Change Mohamed To Ahmed
?>
```

اضافة عنصر جديد : يمكنك بكل بساطة أن تضيف عنصر جديد للمصفوفة عن طريق تحديد مفتاح للعنصر الجديد بين الأقواس المربعة [ ] ومن ثم تعيين القيمة الأساسية بأستخدام علامة التعيين . (=)

```
<?php
$person = array("name" => "Mohamed","adress" => "green","age" => "blue");
$person["tall"] = "160 cm" ; // اضافة قيمة جديدة للمصفوفة
?>
```

"tall" => "160cm"),

$ person = array(" name " => " Ahmed",

مصفوفات متداخلة Multidimensional Arrays

المصفوفات المتداخلة عبارة عن مصفوفة داخل مصفوفة أخرى، ويمكننا اضافة مصفوفة داخل مصفوفة بدلاً عن العنصر العادي، ويتم التحكم في المصفوفات المتداخلة عن طريق سهلة جداً,

[رقم العنصر في المصفوفة الفرعية] [رقم العنصر في المصفوفة الرئيسية]

```
<?php
$person = array("Mohamed","Khaled","Osman",array("ibrahim","abdo","Eslam"));
echo $person [3][0] // ibrahim
echo $person [3][1] // abdo
?>
```

الدوالFunction

الدالة هي جزء من كود البرنامج يتم تعريفه عن طريق المبرمج ليتم تنفيذ شي معين بواسطتها، حيث تقوم الدالة بأخذ قيم وتسمي arguments (معطيات) كمدخلات، ثم تقوم بعمل بعض التعديلات على هذه المدخلات لتقوم بإخراج قيم أخرى... وفي أكثر الأحيان تقوم الدالة بأخذ القيم و وضعها في متغيرات أخري تسمي بالـ (parameters) حتى يتم إجراء العمليات عليها داخل الدالة وهذه المتغيرات لا تعمل خارج الدالة أي أنها متغيرات خاصة بالدالة فقط!!!

الدوالFunction

كما ذكرنا سابقاً فإنه يوجد في لغة الـ " PHP " العديد من الدوال Function والتي تقوم بوظيفة معينة (محددة) ومن فوائد الدوال تصغير الكود البرمجي والتخلص من تكرار بعض المهام التي تتكرر باستمرار، وكذلك توجد إمكانية إنشاء دوال تؤدي وظيفة خاصة وسوف نركز في هذا الفصل من الكتاب على هذا النوع من الدوال الخاصة (كيفية إنشاء دوال) والتي تقوم بتنفيذ شيء معين حيث تأخذ (متغيرات – معطيات) ثم تقوم بمعالجة هذه المتغيرات و تخرج قيمة أخرى.

تعريف الدالة:

لكي تقوم بتعريف دالة فإننا نقوم بكتابة الكلمة " function " متبوعة باسم الدالة والبارمترات اللازمة والتي سوف يتم إجراء العمليات عليها بين قوسين ومن ثم تقوم بكتابة الكود اللازم وسط القوسين المعقوفين . " { } "

و الصيغة العامة للدالة هي:

```
Function function_name (parameters)
{
function code
}
```

ولاحظ أن الكود السابق يتكون من عدة أجزاء :

1- Function إن أي دالة يجب أن تبدأ بهذه الكلمة وهي تشير إلى أن الكود اللاحق له عبارة عن الدالة التي نريد تعريفها .

2- function_name أو أسم الدالة التي سوف نقوم بتعريفها، ونستطيع أن نحدد أي اسم باستثناء الدوال المعرفة فعلاً، ولكن يفضل أن يكون هذا الاسم يدل على معنى الدالة، حتى يسهل تذكرها عند استدعاءها فيما بعد .

3- الجزء الخاص بعوامل الدالة والذي يحتوي على البارامتر أو المتغيرات .

4- {} كود الدالة بين القوسين المعقوفين، وكل ما يكتب بعدها لن يتم تنفيذه مباشرة، بل سيتم ذلك عند استدعاء الدالة .

5- return وذلك لإعلام الدالة بأن وظيفتها قد انتهت بالإضافة إلى ذكر اسم المتغير المذكور في

تعريف الدالة سابقاً .

وعند برمجة الدالة نقوم بكتابه اسم الدالة بدلاً من " function_name " ثم نقوم بتعريف المتغيرات " parameters "، ثم نقوم بكتابه الكود الذي سوف يقوم بالمطلوب بين القوسين بدلاً من " function code ".

ملاحظات على الدوال:

1- أن تكون الدالة غير موجودة مسبقاً، لأنها لو كانت موجودة فعلاً فليس هناك سبب يدعوا لتكرار الجهد، بل يمكنك إستخدام الدوال الجاهزة في دليل الـ php.net .

2- لا يمكن أن يبدأ اسم الدالة بعدد .

3- عدم وضع فاصلة منقوطة بعد نهاية الدالة.

والآن هيا نقوم بكتابة دالة من إنشاءنا والتي تقوم بإجراء عملية الجمع على متغيرين وسوف نقوم بتسمية الدالة باسم " sum " وهو اسم قمنا باختياره وهو يدل على وظيفة وهدف الدالة ويجب أن تعلم أنه يمكنك أن تقوم بتسمية الدالة بأي اسم تريده ولست مجبراً على اختيار اسم معين.

```
<?
Function sum($a)
{
$a = $a + 100 ;
return $a;
}
?>
```

الشرح:

تقوم في هذه الدالة بإجراء عملية إضافة 100 على المتغير أو القيمة التي يتم تمريرها.

Return

يجب أن نضعها في نهاية كل دالة، نستخدم هذه الكلمة لكي نقوم بإعلام الدالة أن وظيفتها انتهت، وأيضاً نستخدمها إذا كان لدينا أكثر من قيمة ونريد أن نقوم بإخبار الـ " PHP " ما هي القيمة التي سيتم اعتمادها ففي مثالنا هذا أردنا إخبار الـ " PHP " بأن يقوم بأخذ المتغير $a على أنه هو القيمة النهائية، حتى لو لم نضع المتغير فسيتم اعتباره هو الناتج النهائي لأنه لا يوجد متغير آخر تم عليه أي عمليات.

مثال:

```
<?
Function sum($a)
{
$a = $a + 100 ;
return $a ;
}
```

```
?>
```

لا توجد مشاكل هنا لأنه لا يوجد لدينا إلا قيمة واحدة لن يتم اعتماد قيمة غيرها ولكن على افتراض أنه لدينا أكثر من قيمة كما في المثال التالي:

```
<?
Function sum($a,$b)
{
$a = $a + 100 ;
$b= $b*100;
return $a ;
}
?>
```

هنا يجب تحديد أي المتغيرين سيكون هو القيمة النهائية للدالة.

شرح الدالة:

تقوم الدالة التي صنعناها بأخذ قيمتين ومن ثم فإنها تقوم بزيادة العدد الذي يتم تمريره 100، ولكي نقوم بإخراج نتيجة الدالة فإننا ببساطة نستطيع ذلك بإجراء أحد الأمرين echo أو. print

مثال:

```
<?
Function sum($a)
{
$a = $a + 100 ;
return ;
}
echo sum(500); ?>
```

لقد قمنا بتمرير رقم بدلاً من المتغير ويمكننا أيضا تمرير متغير بدلاً من الرقم

مثال:

```
<?
Function sum($a)
{
$a = $a + 100 ;
return ; → $a
}
$f=100;
echo sum($f);
```

```
?>
```

لاحظ أننا استخدمنا متغير في الدالة (مما يثبت كلامنا في الأعلى أن للدالة متغيرات خاصة بها) وليس معني ذلك أننا لا نستطيع استخدام متغيرات بنفس الاسم المذكور في الدالة فيمكننا مثلاً كتابة نفس اسم المتغير بدون حصول أي مشاكل كالتالي:

```
<?
Function sum ($a)
{
$a = $a + 100 ;
return ;
}
$a=100;
echo sum($a);
?>
```

كما يمكننا أيضا استدعاء دالة بشكل عادي إذا كانت هي تقوم بالطباعة.

مثال:

```
<?
Function sum($a)
{
$a = $a + 100 ;
print $a;
return ;
}
$a=100;
sum($a); ?>
```

print

يقوم الأمر " print " بنفس عمل الدالة " echo " ولا يوجد بينهما أي اختلاف سوى أن الدالة " echo "قديمة وهي الأصل أما الدالة " print " فقد تم إنشاؤها في." php4 "

مثال:

```
<?
// تعريف الدالة
function ahmed($a)
{
$a=$a*$a*$a*$a;
```

```
return($a);
}
// طبـاعة ناتج الدالة عند إدخال الرقم 5 فيها //
echo ahmed(5);
?>
```

الشرح:

هذه الدالة تقوم بحساب عدد مرفوع لأس أربعة بمعنى أن العدد مضروب في نفسه أربع مرات اسم الدالة ahmed ، وعند طباعة مخرجات الدالة لرقم ، كتبنا أمر الطباعة قبل اسم الدالة والرقم المراد حساب الأس الرابع له بين قوسين (5) و هكذا إذا وضعنا أي رقم آخر سوف تقوم الدالة بحساب الأس الرابع للرقم مباشر و في مثالنا هذا يتم طبع الرقم 625 .

تمرير القيم إلى الدالة :

يمكن القول بأن هناك نوعين من تمرير القيم .

1- تمرير القيمة مباشرة إلى الدالة. (passing by value)

و ذلك بوضع القيمة مباشرة بدون إدراجها في متغيرات.

مثال:

```
<?
Function tiba ($f)
{
$f=$f+$f;
return ;
}
echo tiba(100);
?>
```

لاحظ أننا قمنا بإدراج القيمة مباشرة للدالة بدون وضعها في متغيرات .

2- تمرير القيمة عن طريق المرجع. (passing by reference)

و يتم ذلك بوضع القيمة في متغير أولاً ، ثم وضع هذا المتغير في الدالة لكي يتم إجراء العمليات عليه.

مثال:

```
<?
Function tiba ($f)
{
$f=$f+$f;
return ;
}
$r =1000;
```

```
echo tiba($r);
?>
```

مثال:

```
<?
function ahmed($a)
{
$a=$a*$a*$a*3 ;
return($a) ;
}
$z=10 ;
echo ahmed ($z) ;
?>
```

الشرح:

في هذا المثال تقوم الدالة بضرب العدد في نفسه ثلاث مرات ثم في الرقم 3 ، و نلاحظ أننا مررنا المتغير z$ إلى الدالة ahmed و كتبناها جميعها في سطر طباعة نتيجة الدالة بالأمر echo . و لذلك تقوم الدالة في هذا المثال بضرب الرقم 10 في نفسه ثلاث مرات ثم في 3 يكون الناتج 3000 و من ثم يتم طباعة الناتج ، و بطبيعة الحال كلما غيرنا قيمة المتغير اختلفت نتيجة الدالة .

إعداد قيمة افتراضيه للدالة .

تستطيع أن تجعل الـ " PHP4 " يقوم بإدراج قيمة افتراضية عند عدم تمرير متغيرات إليه.

مثال:

```
<?
Function tiba ($f=40)
{
$f=$f+$f;
return ;
}
echo tiba();
?>
```

إذا لم يتم إعطاء قيمة للدالة فإنها ستفترض أن القيمة هي 40 مباشرة .

أما إذا تم تمرير قيمة أو متغير فإنه سيتم العمل بالقيمة التي تم تمريرها بدلاً من القيمة الافتراضية.

```
<?
Function tiba ($f=40)
{
$f=$f+$f;
return ;
}
echo tiba(100);
?>
```

مدى المتغيرات :(variable scope)

هناك متغيرات محلية (local) و متغيرات عامة (global) ، و نحن نعني بالمتغيرات المحلية التي تكون في داخل الدالة ، ونعني بالعامة التي تكون في كود الـ " PHP " بشكل عام.

مثال:

```
<?
//هذا متغير عام
$r= "ahmed";
function tiba($s)
{
//هذا متغير محلى
$s = "progrramer";
}
echo $r;    ahmed
tiba($s);
echo $s; ?>    pule
```

مثال:

```
<?
//هذا متغير عام
$r= "ahmed";
function tiba($s)
{
//هذا متغير محلى
$s = "progrramer";
}
echo $r ;
$s=10;
```

```
echo $s;
?>
```

في المثال الأول استطعنا طباعة المتغير " $r " و لم نستطع طباعة المتغير " $s " لأنه محلى (حيث لا يتم تنفيذه إلا داخل الدالة) ، أما إذا أردنا طباعته فإننا يجب أن نطبع ناتج الدالة لكي نحصل عليه (أي أننا لا نستطيع طباعته بشكل مباشر.(

مثال:

```
<?
هذا متغير عام//
$r = "ahmed";
function tiba($s)
{
هذا متغير محلى//
$s = "programmer";
}
استطعنا طباعته بشكل مباشر//
echo $r ;
tiba($s);
يجب استخدام الدالة لكي يتم طباعته//
echo tiba($s);
?>
```

لاحظ أننا حتى لو قمنا بعملية طباعة المتغير من نفس الدالة فالناتج يكون مختلف لأن لكل متغير عالمه الخاص به .
لكي نقوم بجعل المتغير الذي بداخل الدالة متغيراً عاماً فيمكننا ذلك بإحدى الطريقتين التاليتين:

الطريقة الأولى:

```
<?
function tiba($y)
{
echo $y. "<br>";
global $s;
$s = "programmer";
return ;
}
$f =10;
tiba($f);
echo $s;
```

?>

لاحظ أننا عندما استخدمنا " global " في داخل الدالة لكي يتم تعريف أن المتغير متغير عام ، و بعد أن قمنا باستخدام الدالة قامت بطباعة المتغير المراد طباعته ، و من ثم بعد ذلك قامت بتعريف متغير جديد ($s) و هذا المتغير متغير عام لأننا وضعنا قبله الكلمة " global " فاستطعنا طباعته بكل سهوله.

الجامعة الاسمرية الاسلامية

كلية تقنية المعلومات

علوم الحاسوب

الفصل الدراسي /

خريف 2020/2019

مقرر Internet Programming2

محاضرة بعنوان

التعامل مع قواعد البيانات I

(إضافة الجداول و إضافة البيانات وعرضها من الجدول)

أ.صالح البوري

Create_Table

```
<?php

$database['host'] = 'localhost';

$database['username'] = 'root';

$database['userpass'] = '';

$database['name'] = 'as';


$conn_link = mysqli_connect($database['host'],

$database['username'],$database['userpass'],$database['name']) or

die(mysqli_connect_error());


$query = "CREATE TABLE tbl_name(

user_id INT(10) UNSIGNED AUTO_INCREMENT,

name varchar(100) NOT NULL,

lastname varchar(100) NOT NULL,

email varchar(100) NOT NULL,

UNIQUE (name),
```

```
PRIMARY KEY (user_id)   مفتاح رئيسي

)DEFAULT CHARACTER utf8SET COLLATE utf8_general_ci";

if (mysqli_query($conn_link,$query) === true)

{

echo 'تم انشاء الجدول بنجاح';

}

else

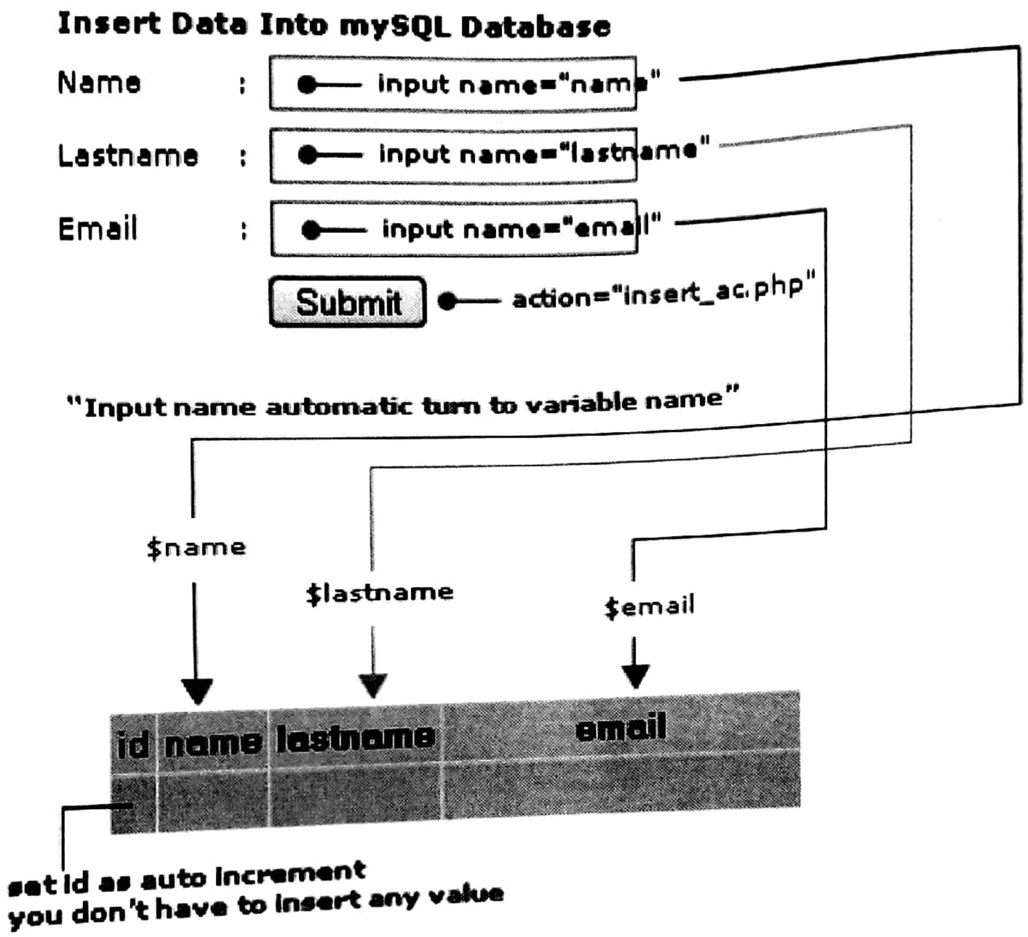
{

echo '<h 3 >   خطا في انشاء الجدول</h3>';

}

?>
```

Insert.php



Insert into mySQL database script

```
$sql="INSERT INTO $tbl_name(name, lastname, email)
VALUES('$name', '$lastname', '$email')";
$result=mysql_query($sql);
```

Insert_ac.php

```
<?php

$database['host'] = 'localhost';

$database['username'] = 'root';

$database['userpass'] = '';

$database['name'] = 'as';


$conn_link = mysqli_connect($database['host'],

$database['username'],$database['userpass'],$database['name']) or

die(mysqli_connect_error());


$name=$_POST['name'];

$lastname=$_POST['lastname'];

$email=$_POST['email'];


$sql="INSERT INTO
tbl_name(name,lastname,email)VALUES('$name','$lastname','$email')";
```

```
$result=mysqli_query($conn_link,$sql) or die ("error55");

if($result){

echo "Successful";

echo "<BR>";

echo "<a href='insert.php'>Back to main page</a>";

}

else {

echo "ERROR";

}

?>
```

Select.php

```
<?php

$database['host'] = 'localhost';

$database['username'] = 'root';

$database['userpass'] = '';

$database['name'] = 'as';


$conn_link = mysqli_connect($database['host'],

$database['username'],$database['userpass'],$database['name']) or

die(mysqli_connect_error());


$sql="SELECT * FROM tbl_name";

$result=mysqli_query($conn_link,$sql) or die ("error55");

?>

<table width="400" border="1" cellspacing="0" cellpadding="2">

<tr>

<td colspan="3"><strong>List data from mysql </strong> </td>

</tr>
```

```
<tr>

<td align="center"><strong>Name</strong></td>

<td align="center"><strong>Lastname</strong></td>

<td align="center"><strong>Your Email</strong></td></tr>

<?php

while($rows=mysqli_fetch_array($result)){

?>

<tr>

<td><?php echo $rows['name']; ?></td>

<td><?php  echo $rows['lastname']; ?></td>

<td><?php  echo $rows['email'];} ?></td></tr></table>
```

الجامعة الاسمرية الاسلامية
كلية تقنية المعلومات
قسم علوم الحاسوب

العام الجامعي 2019
الفصل الدراسي /
ربيع 2018 / 2019

مقرر Internet Programming 2

الفصل الرابع

محاضرة بعنوان :

التعامل مع قواعد البيانات II
(الحذف ،التعديل وترتيب بيانات الجدول)

أ.صالح البوري

Deleteform.php

```
<
table width="300" border="0" align="center" cellpadding="0" cellspacing="1"
bgcolor="#CCCCCC">

<tr>

<"form name="form1" method="post" action="delete.php>

<td>

<table width="100%" border="0" cellpadding="3" cellspacing="1"
bgcolor="#FFFFFF">

<tr>

<td colspan="3"><strong>delete </strong></td>

<tr/>

<tr>

<td width="78">Username</td>

<td width="6">:</td>

<td width="294"><input name="id" type="text" id="id"></td>

<tr/>


<td> </td>

<td> </td>

<td><input type="submit" name="Delete" value="Login"></td>
```

<tr/>

<table/>

<td/>

<form/>

<tr/>

<table/>

Delete.php

```
<?php

$database['host'] = 'localhost';

$database['username'] = 'root';

$database['userpass'] = '';

$database['name'] = 'as';

$conn_link = mysqli_connect($database['host'],

$database['username'],$database['userpass'],$database['name']) or

die(mysqli_connect_error());

$id=$_POST['id'];

$sql="DELETE FROM tbl_name WHERE user_id='$id'" ؛

$result=mysqli_query($conn_link,$sql) or die ("error55؛")

if($result){

echo "Deleted Successfully";

echo "<BR>";

echo "<a href='deleteform.php'>Back to  page delete</a>     " ;

}
```

Updateform.php

```
<table width="300" border="0" align="center" cellpadding="0" >
"cellspacing="1" bgcolor="#CCCCCC>

<tr>

<"form name="form1" method="post" action="update.php>

<td>

<table width="100%" border="0" cellpadding="3" cellspacing="1"
bgcolor="#FFFFFF>


<tr>

<td colspan="3"><strong>delete </strong></td>

<tr/>

<tr>

<td width="78">update via Id number</td>

<td width="6">:</td>

<td width="294"><input name="id" type="text" id="id"></td>

<tr/>


<tr>

<td width="78">update name</td>

<td width="6">:</td>
```

```
<td width="294"><input name="name" type="text" id="name"></td>

<tr/>

<tr>

<td>update lastname</td>

<td>:</td>

<td><input name="lastname" type="text" id="lastname"></td>

<tr/>


<tr>

<td> update your Email</td>

<td>:</td>

<td><input name="email" type="text" id="email"></td>

<tr/>

<tr>

<td> </td>

<td> </td>

<td><input type="submit" name="Delete" value="Login"></td>

<tr/>

<table/>

<td/>

<form/>

<tr/>

<table/>
```

Update.php

```
<? Php

$database['host'] = 'localhost';

$database['username'] = 'root';

$database['userpass'] = '';

$database['name'] = 'as';

$conn_link = mysqli_connect($database['host'],

$database['username'],$database['userpass'],$database['name']) or

die(mysqli_connect_error());

id=$_POST['id'];$


$name=$_POST['name'];

$lastname=$_POST['lastname'];

$email=$_POST['email'];

sql="UPDATE tbl_name SET name='$name', lastname='$lastname',
email='$email' WHERE user_id='$id'  ";

$result=mysqli_query($conn_link,$sql) or die ("error55");
```

```
if($result)
{
echo "update Successfully" ;
echo "<BR>";
echo "<a href='updateform.php'>Back to  page delete</a>";
}
else

{
echo "ERROR";
}


?>
```

أ.صالح البوري

أكواد ترتيب بيانات الجدول

"SELECT column_name FROM table_name **ORDER BY column_name ASC**";

or

"SELECT column_name FROM table_name";

- This will select the records from mysql by ascending (ascending is the default value).

"SELECT column_name FROM table_name **ORDER BY column_name DESC**";

- This will select the records from mysql by descending